Practical-1

Q1) What is android?

Android is an open-source and Linux-based operating system. It was first introduced on Nov 5, 2007. It was originally developed by Android Inc. and subsequently purchased by Google.

Basically, Android is thought of as a mobile operating system. But it is not limited to mobile-only. It is currently used in various devices such as mobiles, tablets, televisions etc.

Android provides a rich application framework that allows us to build innovative apps and games for mobile devices in a Java language environment.

The Android open-source software stack consists of Java applications running on a Java-based, object-oriented application framework on top of Java core libraries running on a Dalvik virtual machine featuring JIT compilation.

Android Studio is the official integrated development environment (IDE) for Android application development. It is based on the IntelliJ IDEA, a Java integrated development environment for software, and incorporates its code editing and developer tools.

To support application development within the Android operating system, Android Studio uses a Gradle-based build system, emulator, code templates, and Github integration. Every project in Android Studio has one or more modalities with source code and resource files. These modalities include Android app modules, Library modules, and Google App Engine modules.

Android Studio uses an Instant Push feature to push code and resource changes to a running application. A code editor assists the developer with writing code and offering code completion, refraction, and analysis. Applications built in Android Studio are then compiled into the APK format for submission to the Google Play Store.

The software was first announced at Google I/O in May 2013, and the first stable build was released in December 2014. Android Studio is available for Mac, Windows, and Linux desktop platforms. It replaced Eclipse Android Development Tools (ADT) as the primary IDE for Android application development. Android Studio and the Software Development Kit can be downloaded directly from Google.

Download Link for Android SDK :- https://developer.android.com

Q2) Explain Manifest file.

AndroidManifest.xml file in android

The **AndroidManifest.xml file** *contains information of your package*, including components of the application such as activities, services, broadcast receivers, content providers etc.

It performs some other tasks also:

- It is **responsible to protect the application** to access any protected parts by providing the permissions.
- It also **declares the android api** that the application is going to use.
- It **lists the instrumentation classes**. The instrumentation classes provides profiling and other informations. These informations are removed just before the application is published etc.

This is the required xml file for all the android application and located inside the root directory.

A simple AndroidManifest.xml file looks like this:

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.javatpoint.hello"
    android:versionCode="1"
    android:versionName="1.0" >

    <uses-sdk
        android:minSdkVersion="8"
        android:targetSdkVersion="15" />

    <application
        android:icon="@drawable/ic_launcher"
        android:label="@string/app_name"
        android:theme="@style/AppTheme" >
        <activity
            android:name=".MainActivity"
            android:label="@string/title_activity_main" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
```

```
22.         </activity>
23.     </application>
24.
25. </manifest>
```

Elements of the AndroidManifest.xml file

The elements used in the above xml file are described below.

<manifest>

manifest is the root element of the AndroidManifest.xml file. It has **package** attribute that describes the package name of the activity class.

<application>

application is the subelement of the manifest. It includes the namespace declaration. This element contains several subelements that declares the application component such as activity etc.

The commonly used attributes are of this element are **icon**, **label**, **theme** etc.

android:icon represents the icon for all the android application components.

android:label works as the default label for all the application components.

android:theme represents a common theme for all the android activities.

<activity>

activity is the subelement of application and represents an activity that must be defined in the AndroidManifest.xml file. It has many attributes such as label, name, theme, launchMode etc.

android:label represents a label i.e. displayed on the screen.

android:name represents a name for the activity class. It is required attribute.

<intent-filter>

intent-filter is the sub-element of activity that describes the type of intent to which activity, service or broadcast receiver can respond to.

<action>

It adds an action for the intent-filter. The intent-filter must have at least one action element.

<category>

It adds a category name to an intent-filter.

Q3) Explain R file.

Android R.java is *an auto-generated file by aapt* (Android Asset Packaging Tool) that contains resource IDs for all the resources of res/ directory.

If you create any component in the activity_main.xml file, id for the corresponding component is automatically created in this file. This id can be used in the activity source file to perform any action on the component.

Note: If you delete R.jar file, android creates it automatically.

Let's see the android R.java file. It includes a lot of static nested classes such as menu, id, layout, attr, drawable, string etc.

```
1. /* AUTO-GENERATED FILE. DO NOT MODIFY.
2.  *
3.  * This class was automatically generated by the
4.  * aapt tool from the resource data it found.  It
5.  * should not be modified by hand.
6.  */
7.
8. package com.example.helloandroid;
9.
10. public final class R {
11.     public static final class attr {
12.     }
13.     public static final class drawable {
14.         public static final int ic_launcher=0x7f020000;
15.     }
16.     public static final class id {
17.         public static final int menu_settings=0x7f070000;
18.     }
19.     public static final class layout {
20.         public static final int activity_main=0x7f030000;
21.     }
22.     public static final class menu {
23.         public static final int activity_main=0x7f060000;
24.     }
25.     public static final class string {
26.         public static final int app_name=0x7f040000;
27.         public static final int hello_world=0x7f040001;
28.         public static final int menu_settings=0x7f040002;
29.     }
30.     public static final class style {
31.         /**
32.         Base application theme, dependent on API level. This theme is replaced
```

```
33.         by AppBaseTheme from res/values-vXX/styles.xml on newer devices.
34.
35.
36.             Theme customizations available in newer API levels can go in
37.             res/values-vXX/styles.xml, while customizations related to
38.             backward-compatibility can go here.
39.
40.
41.         Base application theme for API 11+. This theme completely replaces
42.         AppBaseTheme from res/values/styles.xml on API 11+ devices.
43.
44. API 11 theme customizations can go here.
45.
46.         Base application theme for API 14+. This theme completely replaces
47.         AppBaseTheme from BOTH res/values/styles.xml and
48.         res/values-v11/styles.xml on API 14+ devices.
49.
50. API 14 theme customizations can go here.
51.          */
52.         public static final int AppBaseTheme=0x7f050000;
53.         /** Application theme.
54.    All customizations that are NOT specific to a particular API-level can go here.
55.    */
56.         public static final int AppTheme=0x7f050001;
57.      }
58. }
```

Q4) Significance of activity_main.xml and mainactivity.java

activity_main.xml

The **activity_main.xml** is a layout file available in res/layout directory, that is referenced by application when building its interface. We will modify this file very frequently to change the layout of application.

For "Hello World!" application, this file will have following content related to default layout

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
  xmlns:tools="http://schemas.android.com/tools"
  android:layout_width="match_parent"
  android:layout_height="match_parent" >

  <TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_centerHorizontal="true"
    android:layout_centerVertical="true"
    android:padding="@dimen/padding_medium"
    android:text="@string/hello_world"
    tools:context=".MainActivity" />

</RelativeLayout>
```

The TextView is an Android control used to build the GUI and it have various attributes like android:layout_width, android:layout_height etc which are being used to set its width and height etc.. The @string refers to the strings.xml file located in the res/values folder. Hence, @string/hello_world refers to the hello string defined in the strings.xml file, which is "Hello World!".

MainActivity.java

The main activity code is a Java file **MainActivity.java**. This is the actual application file which ultimately gets converted to a Dalvik executable and runs your application.

Following is the default code generated by the application wizard for "HelloWorld!"

```
package com.example.helloworld;

import android.support.v7.app.AppCompatActivity;

import android.os.Bundle;


public class MainActivity extends AppCompatActivity {
  @Override
  protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
  }
}
```

R.layout.activity_main refers to the activity_main.xml file located in the res/layout folder. The onCreate() method is one of many methods that are figured when an activity is loaded.

Q5) Create a simple app of button with output.

MainActivity.java

```
package com.example.pract1;

import android.annotation.SuppressLint;
import android.os.Bundle;
import android.view.View;
import android.widget.TextView;

import androidx.appcompat.app.AppCompatActivity;

public class MainActivity extends AppCompatActivity {

   private TextView tv1;

   @Override
   protected void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);
      setContentView(R.layout.activity_main);
      tv1=findViewById(R.id.textView);
   }

   @SuppressLint("SetTextI18n")
   public void Text(View v){
      tv1.setText("You Clicked a Button");
   }
}
```

string.xml

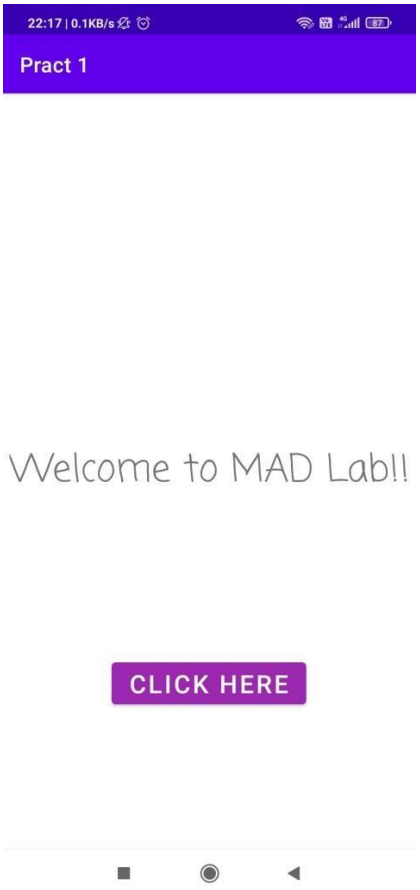
```
<resources>
   <string name="app_name">Pract 1</string>
   <string name="Welcome_to_Mad_lab">Welcome to MAD Lab!!</string>
   <string name="Button">Click Here</string>
   <string name="New_text">You Clicked a Button</string>
</resources>
```

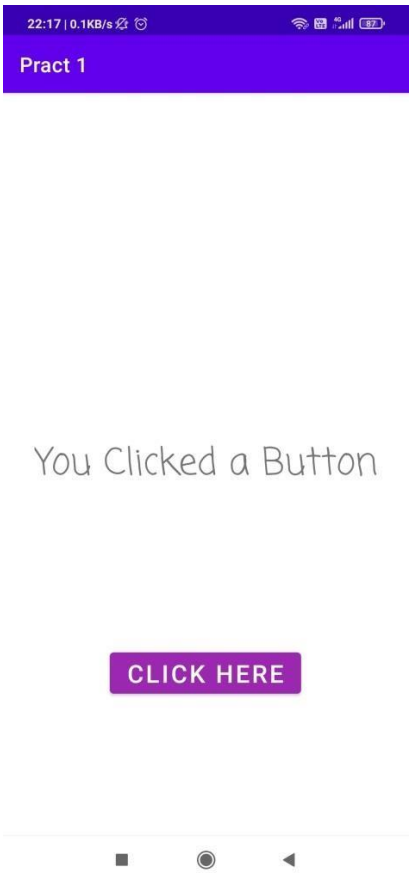
Activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:app="http://schemas.android.com/apk/res-auto"
   xmlns:tools="http://schemas.android.com/tools"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
```

```
    tools:context=".MainActivity">

    <TextView
        android:id="@+id/textView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:fontFamily="casual"
        android:text="@string/Welcome_to_Mad_lab"
        android:textSize="36sp"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        tools:visibility="visible" />

    <Button
        android:id="@+id/button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:onClick="Text"
        android:text="@string/Button"
        android:textSize="24sp"
        app:backgroundTint="#9C27B0"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.497"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/textView"
        app:layout_constraintVertical_bias="0.531"
        tools:ignore="UsingOnClickInXml" />

    <TextView
        android:id="@+id/textView2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:fontFamily="casual"
        android:text="@string/New_text"
        android:textSize="34sp"
        android:visibility="invisible"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/button" />

</androidx.constraintlayout.widget.ConstraintLayout>
```

Output:-


22:17 | 0.1KB/s
Pract 1
Welcome to MAD Lab!!
CLICK HERE


22:17 | 0.1KB/s
Pract 1
You Clicked a Button
CLICK HERE